How to Modify Themes in VSCode

3 min read

- #DevTools

Ever found a theme you really liked but there were just a few small blemishes that kept gnawing at you?

It happened to me with the Monokai theme that comes with VSCode. But fortunately, VSCode allows you to modify almost every single aspect of a theme.

There are two types of customization available in VSCode.

- workbench.colorCustomizations
- editor.tokenColorCustomizations

workbench.colorCustomizations

This allows you to customize the VSCode interface. You can customize the sidebars, various backgrounds, status bar, mini-map and much more. I mainly used it to tune the colors to increase contrast.

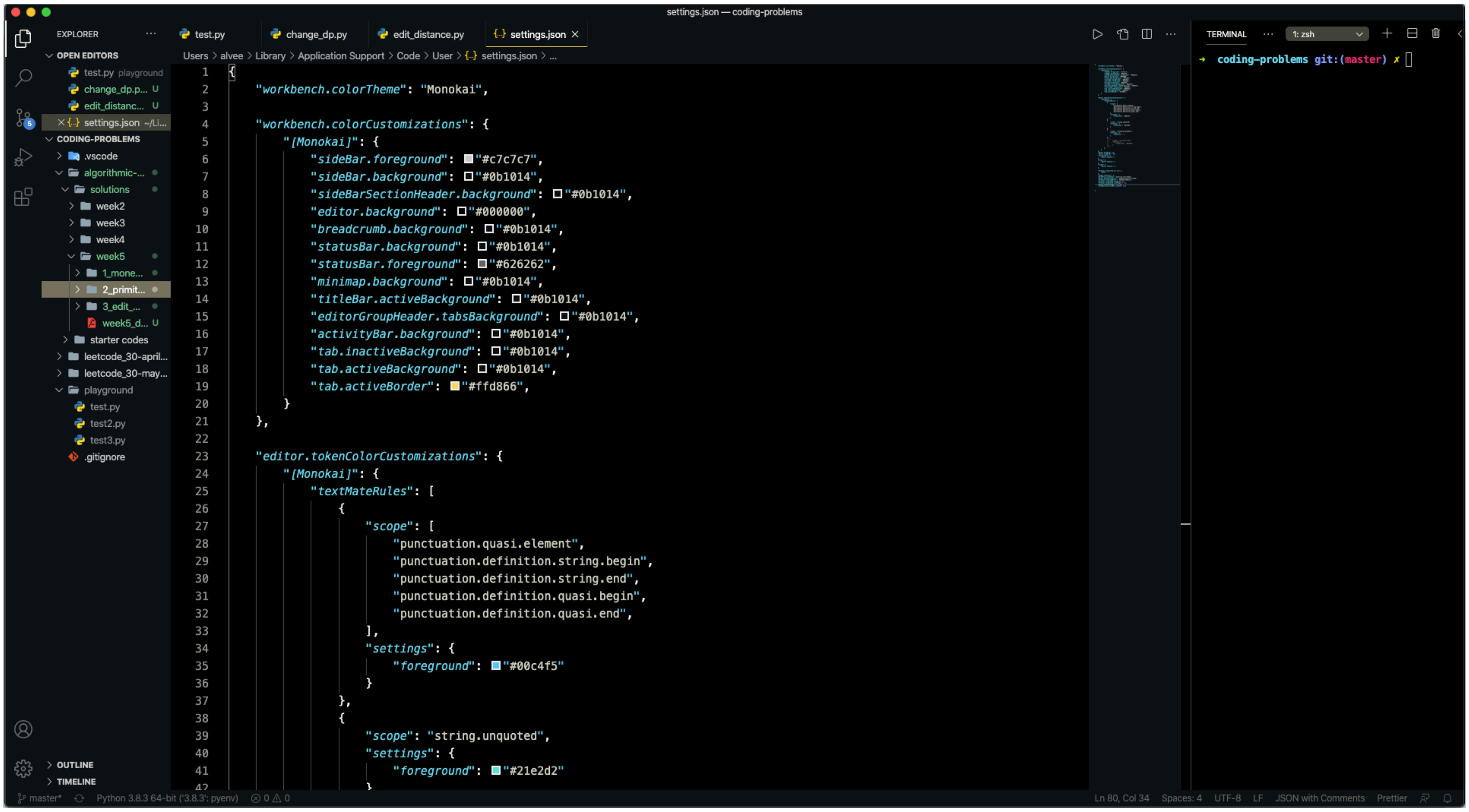
Here’s how to access and customize it:

- Open the user settings by hitting `ctrl + shift + P` and searching for `Preferences: Open Settings (JSON)`.
- Create the `workbench.colorCustomization` JSON property if it’s not already created.
- Create a nested property with your theme’s name in `[]` such as `[Monokai]`.
- Create an object.
- Add and modify the properties you want.

Check out the full list of options here.

Here’s how mine looks with the customizations for the Monokai theme.

```
"workbench.colorCustomizations": {
    "[Monokai]": {
        "sideBar.foreground": "#c7c7c7",
        "sideBar.background": "#0b1014",
        "sideBarSectionHeader.background": "#0b1014",
        "editor.background": "#000000",
        "breadcrumb.background": "#0b1014",
        "statusBar.background": "#0b1014",
        "statusBar.foreground": "#626262",
        "minimap.background": "#0b1014",
        "titleBar.activeBackground": "#0b1014",
        "editorGroupHeader.tabsBackground": "#0b1014",
        "activityBar.background": "#0b1014",
        "tab.inactiveBackground": "#0b1014",
        "tab.activeBackground": "#0b1014",
        "tab.activeBorder": "#ffd866",
    }
}
```



```
{
    "workbench.colorTheme": "Monokai",

    "workbench.colorCustomizations": {
        "[Monokai]": {
            "sideBar.foreground": "#c7c7c7",
            "sideBar.background": "#0b1014",
            "sideBarSectionHeader.background": "#0b1014",
            "editor.background": "#000000",
            "breadcrumb.background": "#0b1014",
            "statusBar.background": "#0b1014",
            "statusBar.foreground": "#626262",
            "minimap.background": "#0b1014",
            "titleBar.activeBackground": "#0b1014",
            "editorGroupHeader.tabsBackground": "#0b1014",
            "activityBar.background": "#0b1014",
            "tab.inactiveBackground": "#0b1014",
            "tab.activeBackground": "#0b1014",
            "tab.activeBorder": "#ffd866",
        }
    },

    "editor.tokenColorCustomizations": {
        "[Monokai]": {
            "textMateRules": [
                {
                    "scope": [
                        "punctuation.quasi.element",
                        "punctuation.definition.string.begin",
                        "punctuation.definition.string.end",
                        "punctuation.definition.quasi.begin",
                        "punctuation.definition.quasi.end",
                    ],
                    "settings": {
                        "foreground": "#00c4f5"
                    }
                },
                {
                    "scope": "string.unquoted",
                    "settings": {
                        "foreground": "#21e2d2"
                    }
```

```
➜ coding-problems git:(master) ✗
```

editor.tokenColorCustomizations

This lets you modify the syntax highlighting of the theme.

Here are the steps:

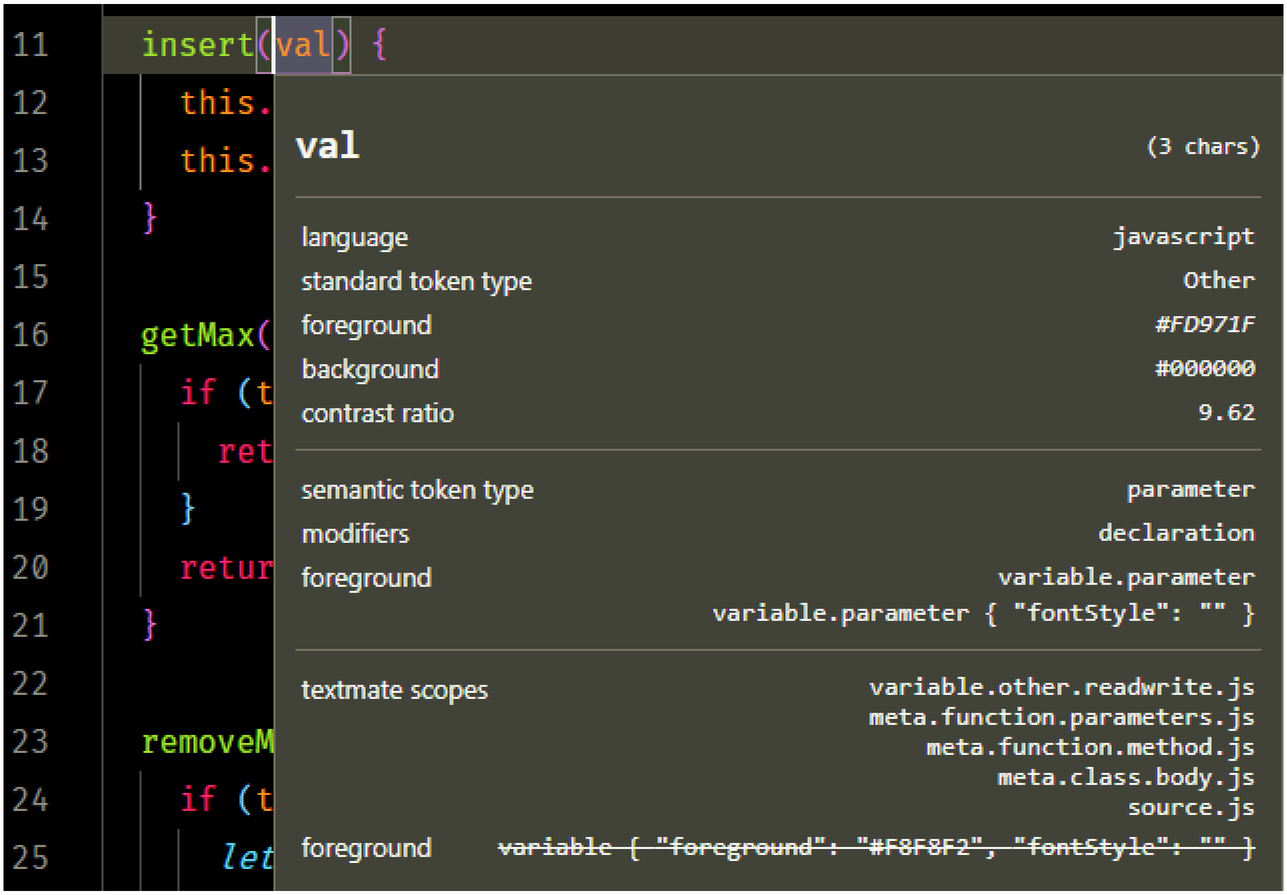
- Open the user settings by hitting `ctrl + shift + P` and searching for `Preferences: Open Settings (JSON)`.
- Create the `editor.tokenColorCustomizations` JSON property if it’s not already created.
- Create a nested property with your theme’s name in `[]` such as `[Monokai]`.
- Create another nested property called `textMateRules` and initialize to an empty array.
- Create objects with `scope` and `settings` keys to modify the defined scopes.

How to Find Out TextMate Scope in VSCode

If you are not familiar with TextMate scopes here’s how to figure them out in VSCode:

- Hit `ctrl + shift + P`.
- Search for `Developer: Inspect Editor Tokens and Scopes` and hit enter.

This will open a window and show you all the details about that scope.



```
insert(val) {
  this.
  this.
}

getMax(
  if (t
    ret
  }
  retur
}

removeM
  if (t
    let
```

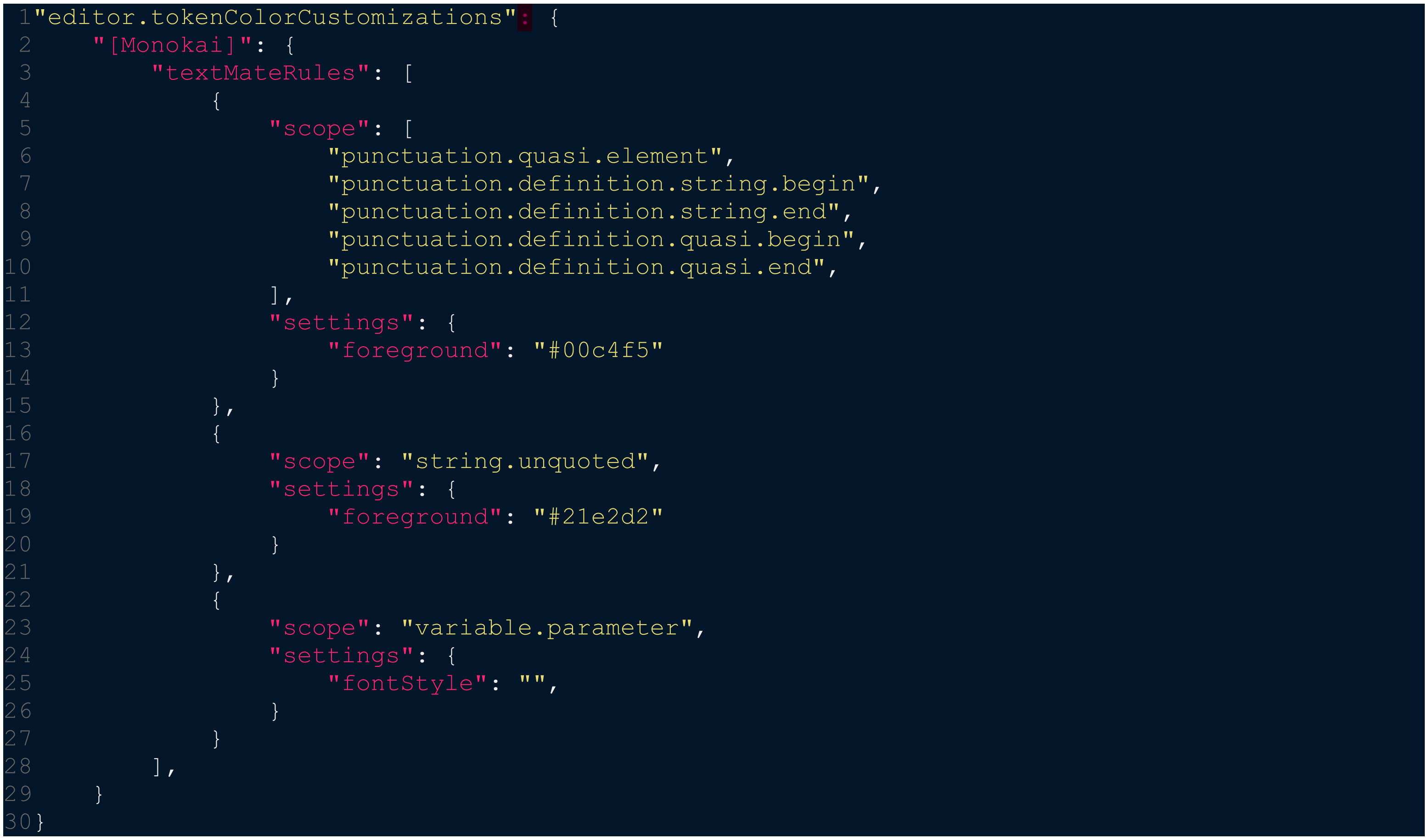
val (3 chars)

language javascript
standard token type Other
foreground #FD971F
background #000000
contrast ratio 9.62

semantic token type parameter
modifiers declaration
foreground variable.parameter
variable.parameter { "fontStyle": "" }

textmate scopes variable.other.readwrite.js
meta.function.parameters.js
meta.function.method.js
meta.class.body.js
source.js
foreground ~~variable { "foreground": "#F8F8F2", "fontStyle": "" }~~

Here’s my code:



```
"editor.tokenColorCustomizations": {
    "[Monokai]": {
        "textMateRules": [
            {
                "scope": [
                    "punctuation.quasi.element",
                    "punctuation.definition.string.begin",
                    "punctuation.definition.string.end",
                    "punctuation.definition.quasi.begin",
                    "punctuation.definition.quasi.end",
                ],
                "settings": {
                    "foreground": "#00c4f5"
                }
            },
            {
                "scope": "string.unquoted",
                "settings": {
                    "foreground": "#21e2d2"
                }
            },
            {
                "scope": "variable.parameter",
                "settings": {
                    "fontStyle": "",
                }
            }
        ],
    }
}
```